CLIENT-SERVER SEMICONDUCTOR VERIFICATION SYSTEM

FIELD OF THE INVENTION

[0001] The present invention relates generally to a semiconductor design verification system, and in particular, to a system for and method of verifying a digital semiconductor design.

BACKGROUND OF THE INVENTION

[0002] Circuit design verification can require significant time and costly equipment, including both software and hardware. Currently, a behavioral description and test bench in hardware description language (HDL) are created for a design of a semiconductor device. Even if a behavioral description is not created, the design is coded in register transfer level (RTL) and simulated using a testbench. When the design is implemented in a target architecture, lengthy simulations using timing and functional models of the target architecture are run. As designs and technology are becoming more complex, the time required to run these simulations is becoming increasingly long. Although hardware accelerators and hardware emulators are helpful in verifying the design of a semiconductor device, these systems are typically quite expensive. In particular, licenses for software simulators and accompanying hardware peripherals can be prohibitively expensive.

[0003] Some of the high cost of these systems comes from the complexity of mapping designs onto the target architecture, especially for emulation systems. Although hardware accelerators are cheaper than emulators because they carry out simulation loading in vendor specific implementation models including timing information, their functionality is limited. That is, the format of the input data is limited to a format specific to the vendor.

[0004] Similarly, hardware prototype systems also have a

number of drawbacks. Hardware prototype systems are usually in the form of logic analysis tools, from which it is possible to generate, apply and sample stimuli with any semiconductor system having digital inputs and outputs via test probes. However, these systems are not well suited for functional verification.

[0005] Accordingly, there is a need for an improved system and method for verifying a semiconductor design.

SUMMARY OF THE INVENTION

[0006] The systems and methods described in the present disclosure relate to a client/server semiconductor and prototype verification system. The systems provide an inexpensive means for generating and running test vectors at high speed on actual hardware compared to running lengthy software simulations for the purpose of design and prototype functional verification. A test job could be generated in a software environment on a client system. The test job would preferably contain test vectors that would be applied to a semiconductor System Under Test (SUT). The test vectors may be generated using dedicated software or via an API or scripting technique, or determined based upon software behavioral simulations of their design running on external simulation tools. Expected result vectors, which can also be extracted from the results of the software simulation, can be used as reference vectors.

[0007] The systems of the various embodiments of the present invention can be used to create vectors either from their software simulation environment or via alternative methods, and preferably apply these vectors to an unmodified design running on the actual target technology and sample the targets response. Such targets can be any digital system with inputs and outputs. Some specific examples of targets include programmable logic devices (PLD) such as field programmable gate arrays (FPGA).

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] Fig. 1 is a block diagram of a system for verifying a semiconductor design according to an embodiment of the present invention;

[0009] Fig. 2 is a block diagram of a system for verifying a semiconductor design according to an alternate embodiment of the present invention;

[0010] Fig. 3 is a system under test card employed by a system for verifying a semiconductor design according to an embodiment of the present invention;

[0011] Fig. 4 is a system under test card employed by a system for verifying a semiconductor design according to an alternate embodiment of the present invention;

[0012] Fig. 5 is a system under test card employed by a system for verifying a semiconductor design according to another embodiment of the present invention;

[0013] Fig. 6 is a flow chart showing a method of verifying a semiconductor design according to an embodiment of the present invention;

[0014] Fig. 7 is a flow chart showing a method of verifying a semiconductor design according to an alternate embodiment of the present invention; and

[0015] Fig. 8 is a flow chart showing a method of verifying a semiconductor design according to an alternate embodiment of the present invention.

DETAILED DESCRIPTION OF THE DRAWINGS

[0016] The systems and methods of embodiments of the present invention run a software environment on a client system that will create a test job that is sent to a test server. The test server would apply the test job to a semiconductor system under test (SUT) and read back result vectors. The server would then send the result vectors back to the client. It is contemplated that an interactive test system between the client and SUT would exist via the test

server. The SUT can be any semiconductor system or device that can have vectors applied to inputs of the design and have their outputs or test points sampled at discrete times. Although the SUT system finds particular application as a programmable logic device (PLD), the system can be applied to other semi-conductor devices.

[0017] In a design flow, once functional and timing simulations have been completed, development will typically advance to the system prototyping stage. This stage involves running a prototype of the design, in test mode within a real life system, in order to test and evaluate the actual system implementation. The systems of Figs. 1 and 2 provide, in addition to its other features, systems for applying the vectors generated from the behavioral testbench onto the actual inputs of the prototype system, without modification to either testbench or design implementation. The systems provide an extra level of verification, placed between functional verification and real time evaluation testing. According to another aspect of the invention, the test vectors can be run at higher bandwidths, reaching "real design speeds" for the purpose of timing closure or characterization, temperature and power analysis or other environmental analysis of the system running at full speed.

[0018] The systems of Figs. 1 and 2 are inexpensive because only one test server is required. Multiple SUTs can be plugged in and shared among multiple clients from remote locations. SUTs could be developed as dedicated test adapters or could be actual prototype systems. If the target architecture contains PLDs, these can be reconfigured with different target designs for different applications under test. According to another aspect of the present invention, the systems of Figs. 1 and 2 enable vectors to be generated from their existing simulation environments, or they could be generated from external vector generation software.

[0019] Multiple SUTs can be attached to the Test Server

through various technologies, such as peripheral component interconnect (PCI), CoreConnect or an adaptive bus protocol depending on the application and implementation. The test server will execute test jobs on the SUTs and sample their outputs, sending results back to the client system so that the functionality can be evaluated. Some level of interactive debug can occur between client and SUT via the test server. As will be described in more detail in reference to Figs. 3-5, the SUTs can also be dedicated adapters or prototype boards attached to the server via a test bus, or the SUT could be a design attached to a standard bus interface, for example Core IP connected to a CoreConnect Bus. As long as the server is capable of applying digital inputs to and sampling digital outputs from the SUT, then it can be connected using the required technology.

[0020] Referring specifically to Fig. 1, a block diagram of a system for verifying a semiconductor design according to an embodiment of the present invention is shown. In particular, a plurality of client devices 102 are coupled to a server 104 by way of a network 106. The network could be a type of local area network or wide area network commonly employed for enabling communication between computers. The server 104 enables communication with a plurality of systems under test 110. The server 104 preferably comprises a network interface 112 for enabling communication with the network 106 and a system under test interface 116. Finally, the system CPU 114 of the server controls the communication between the network 106 and the systems under test 110.

[0021] The system of Fig. 1 provides a general purpose verification system because any semiconductor System Under Test (SUT) that has digital inputs and outputs can be "plugged" in, and no alteration or additional circuitry/logic is required to the SUT for the purpose of applying test vectors. Also, no proprietary verification language is required for generating the test vectors.

Rather, these vectors can be generated in various ways. One method is the extraction of test vectors from the SUT design behavioral model and testbench. After a design is implemented in RTL, a functionality test is performed against their behavioral model and testbench. Conventional simulations which are carried out by loading in vendor specific implementation models including timing information can take significant time to run. However, the systems according to embodiments of the present invention directly generate test vectors from their behavioral testbench and apply them to an actual SUT. Another method of extracting the vectors to be applied to the SUT is to read the expected outputs from the simulators output, which can be used to compare to the actual response vectors coming back from the system under test.

[0022] Further, the vectors do not need to be applied to the SUT at "real" design speed, which could interfere with functional testing by adding an extra level of complexity (e.g. timing closure). The vectors could be applied at a low frequency for a functionality test. However, it is contemplated that the vectors can be applied close to "real" design speeds with the use of data pumps and specialized circuitry surrounding the SUT, for the purpose of timing characterization. It is also possible to perform power/heat analysis and attach logic analyzers while monitoring the SUT. The SUT could be debugged through the test server, for example by using a purpose built software environment.

[0023] The systems of the embodiments of the present invention are inexpensive because of the simplicity of their implementation. The system is also inexpensive to the end user because only one test server is required to attach multiple SUTs or test adapters. The SUTs or test adapter could be created or purchased separately to plug into any semiconductor verification system. Because the server is detached from the clients, via a LAN/WAN, SUTs can be tested from remote locations. The vectors can be generated from

existing simulation environments or via any alternative method if a simulator is not available. Accordingly, the system does not require a software simulation environment to generate the test vectors.

[0024] The client device will run a software environment on their host computer. The client device will include a method for generating a "test job" that would include stimuli vectors, expected result vectors, timing information and any other relevant information that will be applied to the semiconductor system under test. There are two main ways that the test job can be created. As will be described in more detail below, an interactive graphical user interface (GUI) environment or an external software simulation environment, such as ModelSim from Model Technology Inc. of Wilsonville, OR., could be used.

[0025] The interactive GUI could read in the design or testbench (in HDL or Schematic) and present a waveform construction display. The user could set the values of input signals and preferably set some expected output values. The client device would generate a test job from the data entered via the GUI.

[0026] The behavior of the semiconductor design is typically simulated before full implementation. A behavioral testbench may be created and used to simulate an RTL design. When the simulation is run in an environment such as ModelSim, it is possible to verify that the outputs are as expected. It is possible to extract the values of the input signals and the output signals from the simulation environment and then convert these to test vectors. There are several possible ways of generating test vectors. One way is to have the simulator generate a value change dump (VCD) file which is parsed to get the data. Another way is to use a programming language interface (PLI). It is also possible to insert extra HDL into the testbench to generate the vectors. Finally, it is possible to use scripting languages to generate the vectors, or to use the TCL

programming language to insert a tool into simulation tool to extract vectors.

[0027] Having generated a "test job", the client would send this to the test server. The test server would run the job on the SUT and send results back to the client. The results would then be compared to the expected results and highlight any differences. The client environment could graphically display the results, or the results could be opened in a third party waveform viewer, such as Modeltech's Waveform Viewer. For tighter debug, breakpoints could be set in simulation and the test job could be run in an interactive mode, pausing the execution and stepping through sequences. Although the system is testing a "black box" and the only information returned from the test execution will be the SUT output values, no internal nodes are sampled. However, in order to bring out test points, it is possible to use debug IP, such as Chipscope from Xilinx Inc. of San Jose, CA, or built in self test (BIST) technology if the SUT is a PLD. If the SUT is a circuit board, it is possible to probe test points in the circuit much like a logic analyzer does. It should be noted that the server can be used in an interactive debug mode as well as just sending test jobs.

[0028] The format of the test job itself is not essential to the system, and could come in different formats that the test server would recognize. For example, the simplest format could be a binary file of vectors, or other universal test vector formats. Alternatively, a dedicated format that could be designed so as to be more compact or would be formatted to enable higher throughput to the SUT. The dedicated format could be generated by converting from any universal or binary format.

[0029] Multiple clients can interact with the test server, and the test server can run test jobs on multiple SUTs. The test server will preferably have a microprocessor that will control communication from client devices via TCP as well as the administration of the SUTs and the test jobs,

such as queuing. The SUTs can be physically connected to the test server in a variety of ways. As will be described in more detail in reference to Figs. 3-5, a variety of systems under test can be employed by the system for verifying a semiconductor design.

[0030] The connection between the Server and the SUTs is implemented as a general purpose bus running an adaptive bus protocol. The communication between the Server and the SUT will mainly be concerned with SUT administrating such as in Plug and Play applications, programming configurable devices that are in SUTs, such as PLDs or memories—and coupling of vectors to and from the SUT.

[0031] It is intended that the semiconductor design be unmodified when under test. The test system is designed so that vectors can be applied to the actual inputs of the semiconductor system and the outputs read back from the actual outputs. Therefore, the test system does not insert or require any extra logic or circuitry to the SUT.

[0032] Another embodiment relates to a system having multiple test servers connected via LAN/WAN to one master server or a computer that will act as a job distribution system for all the SUTs connected to the various test servers. SUTs could be accessed from various remote locations where all SUTs are accessed and/or administered from one computer or server. As can be seen in Fig. 2, a block diagram of a system for verifying a semiconductor design according to an alternate embodiment of the present invention is shown. In particular, a plurality of client devices 202 are coupled to a job distribution server 204 by way of a first network 206. The job distribution server 204 is also coupled to a plurality of servers 208 by a second network 210. Finally, a plurality of systems under test 212 are coupled to the servers 208. Although the networks are shown as separate networks, the first and second network could comprise a single network which is accessible by the client devices 202 and the job distribution server 204.

[0033] The systems of some embodiments of the present invention enable a test server test bus using an adaptive protocol running across a general purpose bus on a backplane, with SUT cards attached to the backplane. The SUT cards preferably have a PLD device that is programmed via the backplane and then vectors are applied and read back to the server for the purpose of testing designs running in the PLDs. Each SUT connected could have a different PLD architecture, this would allow designs to be tested across different architectures at high speed. Some SUTs, such as the SUT of Fig. 3, are not "smart" devices, because they only have input vectors applied and outputs sampled. In particular, as shown for example in Fig. 3, a simple SUT adapter 300 comprising a PLD 302 coupled to a connector 304. However, other SUTs are smart because they have a controlling device to communicate with the test server. The test server will detect the SUT when connected and determine what type of SUT it is. If it is a smart SUT, it will know how to communicate with that particular SUT. Different types of 'smart' SUTs can use different communication protocols. In particular, a smart SUT adapter 400 comprising a PLD 402 is coupled to logic registers 404. The logic registers enable decomposition of larger vectors from PLD 402 into smaller vectors which are output by a connector 406. The logic registers also enable the concatenation of smaller vectors from connector 406 to make larger vectors which are coupled to PLD 402.

[0034] In order to run and sample vectors at high speed on the actual SUT adapter, it is unlikely that the Server will be able to transfer vectors at a sufficient rate across the communication bus between the two is employed. It could be possible however to store the vectors in a memory device on the SUT adapter, using the SUT adapter of Fig. 4, for example. The vectors could then be read and applied to the SUT at a higher frequency, while the server works on filling up the vector memory again to the lower test bus speed,

employing a similar technique to that used in data acquisition systems. While these techniques add extra logic they do not interfere with the actual content of the design being verified.

[0035] It may be necessary to measure the behavior of the SUT as well as actually reading its outputs. Examples of this could be power measurement circuits, temperature measurement devices, etc. As was previously mentioned, in order to debug some internal nodes in circuits, or to bring out test signals from internal devices on the board, a logic analysis card could be connected with probes attached to the SUT, as shown for example in Fig. 5. In particular, a PCI prototype board 502 is adapted to be coupled to an SUT 504, as well as a logic analyzer card 506. The PCI prototype board 502 is coupled by way of connector 508 to a connector 510 of the SUT 504. The SUT 504 comprises a logic circuit 512 coupled to a connector 514 which is coupled to a server, such as a server of Figs. 2 or 3. Similarly, a logic analyzer card 506 comprises a logic circuit 518 coupled to a connector 520 which is also coupled to a server.

[0036] Turning now to Fig. 6, a flow chart shows a method of verifying a semiconductor design according to an embodiment of the present invention. In particular, test vectors are generated at a step 602. Expected result vectors are also generated at a step 604. For example, a circuit design may be described in Verilog and a set of test vectors created by the user. A ModelSim simulator may be used to generate the expected result vectors from the Verilog design and the test vectors. The Verilog circuit design may then be synthesized and place and routed using commercially available tools such as provided by Xilinx Inc. of San Jose, CA. The tools generate configuration data that will be used to program the FPGA with the circuit design. A test job having the test vectors and the configuration data are stored in a client device at a step 606. The test server receives the configuration data and configures (or

reconfigures) the system under test, e.g., FPGA, with the circuit design at a step 608. The test vectors are coupled to the system under test, e.g., the circuit design in the FPGA, via the test server at a step 610. An output comprising result vectors of the circuit design is received by the test server from the system under test at a step 612. Finally, the result vectors are coupled to the client device via the test server at a step 614. The result vectors from the system under test are compared to expected result vectors at the client device at a step 616. It is then determined whether the result vectors match the expected vectors at a step 618. If not, the circuit design is modified and new configuration data is generated at a step 620.

[0037] Turning now to Fig. 7, a flow chart shows a method of verifying a semiconductor design according to an alternate embodiment of the present invention. In particular, a plurality of client devices storing test jobs for testing a design are coupled to a server at a step 702. A test server comprising a network interface and a system under test interface is provided a step 704. A plurality of systems under test are coupled to the test server at a step 706. The system under test is reconfigured by way of the test server at a step 708. The test vectors of a predetermined test job are coupled to the system under test by way of the test server at a step 710. An output comprising result vectors are received from the system under test at a step 712. The result vectors are coupled to a client device at a step 714. The result vectors from the system under test are then compared to the expected results vectors at a step 716. It is then determined whether the result vectors match the expected vectors at a step 718. If not, the design is modified and configuration data is generated at a step 720.

[0038] Turning now to Fig. 8, a flow chart shows a method of verifying a semiconductor design according to an

alternate embodiment of the present invention. A plurality of client devices storing test jobs for testing a design of a logic circuit and having test vectors and configuration data for programmable logic are provided at a step 802. A first network is provided between the plurality of client devices and a job distribution server at a step 804. A second network is provided between the job distribution server and a plurality of servers at a step 806. The system under test is reconfigured by way of the test server at a step 808. Predetermined test vectors are coupled to a system under test by way of a server of the plurality of servers at a step 810. An output comprising result vectors from the system under test are received at a step 812. The output comprising result vectors are coupled to a client device at a step 814. Finally, the result vectors from the system under test are compared to expected result vectors at a step 816. It is then determined whether the result vectors match the expected vectors at a step 818. If not, the design is modified and configuration data is generated at a step 820.

[0039] It can therefore be appreciated that the new and novel system for and method of verifying a semiconductor design has been described. It will be appreciated by those skilled in the art that numerous alternatives and equivalents will be seen to exist which incorporate the disclosed invention. As a result, the invention is not to be limited by the foregoing embodiments, but only by the following claims.